SVM

1. Max Margin Classifiers
    - 1.1 Decision boundary / Hyperplanes
    - 1.2 Margin
    - 1.3 objective function
    - 1.4 Support Vectors
2. Support Vector Classifier
    - 2.1 Problem with Max margin classifier
    - 2.2 what is the impact of slack variable
    - 2.3 Advantages
3. SVM
    - 3.1 non-linear Decision Boundary
    - 3.2 kernels
    - 3.3 Advantages
    - 3.4 impact of Gamma and how to tune it
    - 3.5 Metrics

BONUS:

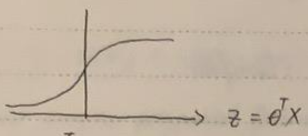
loss function

comparison with logistic regression

SVM 详细整理

Alternative view of logistic regression.

$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$



If $y=1$, we want $h_\theta(x) \approx 1$, $\theta^T x \gg 0$

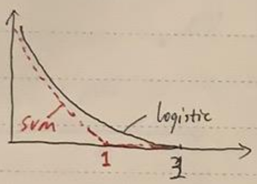
If $y=0$, we want $h_\theta(x) \approx 0$, $\theta^T x \ll 0$

Cost of one example

$$-(y \log h_\theta(x) + (1-y)\log(1-h_\theta(x)))$$

$$= -y \log \frac{1}{1+e^{-\theta^T x}} - (1-y)\log\left(1-\frac{1}{1+e^{-\theta^T x}}\right)$$

If $y=1$ (want $\theta^T x \gg 0$) $\quad Cost = -\cancel{y} \log \frac{1}{1+e^{-\theta^T x}}$ ($y=0$ 同理)



比较 cost function.

Logistic:

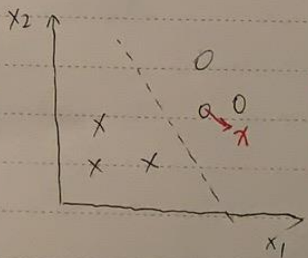
$$\min_\theta \frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \underbrace{(-\log h_\theta(x^{(i)}))}_{Cost_1(\theta^T x^{(i)})} + (1-y^{(i)})\underbrace{((-\log(1-h_\theta(x^{(i)})))}_{Cost_0(\theta^T x^{(i)})}\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

SVM:

$$\min_\theta \cancel{\frac{1}{m}}\, C \sum_{i=1}^{m} y^{(i)} cost_1(\theta^T x^{(i)}) + (1-y^{(i)})\, Cost_0(\theta^T x^{(i)}) + \frac{\cancel{\lambda}}{\cancel{2m}}\sum_{j=1}^{n}\theta_j^2$$

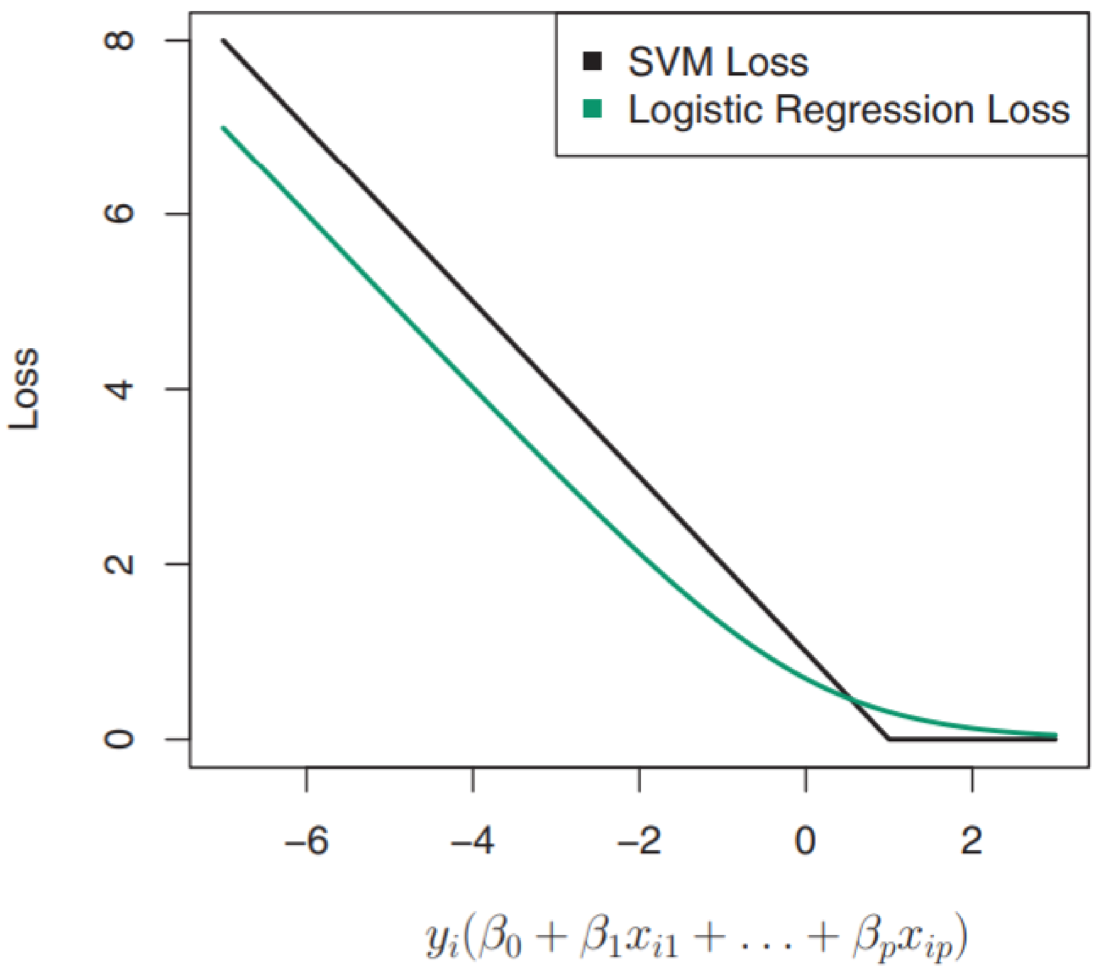
Hypothesis:

$$h_\theta(x) = \begin{cases} 1 & \text{if } \theta^T x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$



θ与x同号,夹角小于90°. 归为一类.

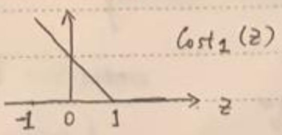
An interesting characteristic of the support vector classifier is that only support vectors play a role in the classifier obtained; observations on the correct side of the margin do not affect it. This is due to the fact that the loss function shown in Figure 9.12 is exactly zero for observations for which $y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq 1$; these correspond to observations that are on the correct side of the margin.[3] In contrast, the loss function for logistic regression shown in Figure 9.12 is not exactly zero anywhere. But it is very small for observations that are far from the decision boundary. Due to the similarities between their loss functions, logistic regression and the support vector classifier often give very similar results. When the classes are well separated, SVMs tend to behave better than logistic regression; in more overlapping regimes, logistic regression is often preferred.



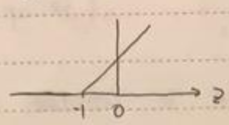
SVM Decision Boundary

$$\min_{\theta} C \underbrace{\sum_{i=1}^{m} \left[y^{(i)} \, Cost_1 (\theta^T x^{(i)}) + (1-y^{(i)}) \, Cost_2 (\theta^T x^{(i)}) \right]}_{*} + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

Whenever $y^{(i)} = 1$: $\theta^T x^{(i)} \geq 1$



whenever $y^{(i)} = 0$ $\theta^T x^{(i)} \leq -1$



So, we transform the things above into another version

I want to fit the parameters to ensure * to be 0.

So, the problem is transformed into

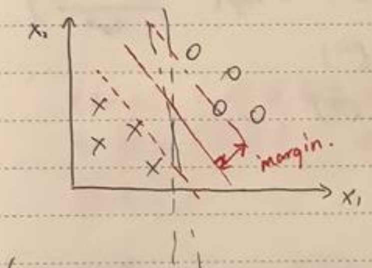
$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2 \quad \text{s.t.} \begin{cases} \theta^T x^{(i)} \geq 1 & \text{if } y^{(i)} = 1 \\ \theta^T x^{(i)} \leq -1 & \text{if } y^{(i)} = 0 \end{cases}$$

$\Longleftrightarrow$

(向量角度解释)

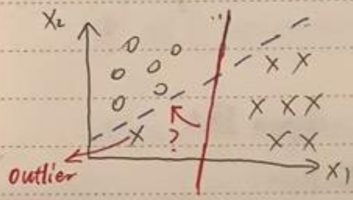
$$\min_{\theta} \frac{1}{2} \|\theta\|^2 \quad \text{s.t.} \begin{cases} p^{(i)} \cdot \|\theta\| \geq 1 & \text{if } y^{(i)} = 1 \\ p^{(i)} \cdot \|\theta\| \leq -1 & \text{if } y^{(i)} = 0 \end{cases}$$

where $p^{(i)}$ is the projection of $x^{(i)}$ onto the vector θ



$\longrightarrow$ Max Margin Classifier

(How does this relate to what I mentioned before? Stay tuned!)

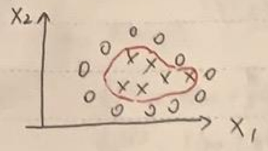


if C very large $\rightarrow$ overfitting

C determines how much we want to avoid misclassifying.

SVM $\rightarrow$ kernel

Non-linear Decision Boundary



predict $y=1$ if $\theta_0+\theta_1x_1+\theta_2x_2+\theta_3x_1x_2+\theta_4x_1^2+\theta_5x_2^2+\cdots \geq 0$

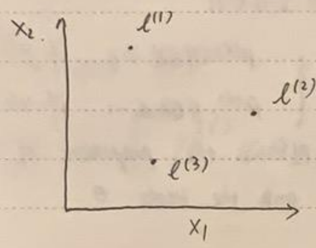
(set up complex polynomial features)

But, is it possible to introduce new notations?

like $\theta_0+\theta_1f_1+\theta_2f_2+\cdots$ to denote new features.

Is there better choice?

$\downarrow$ kernel



Given x, compute new feature depending on proximity to landmarks $\ell^{(1)}, \ell^{(2)}, \ell^{(3)}$.

Define: $f_1 = \text{similarity}(x, \ell^{(1)}) = \exp\left(-\dfrac{\|x-\ell^{(1)}\|^2}{2\sigma^2}\right)$

$f_2 = \text{similarity}(x, \ell^{(2)})$

$f_3 = \text{similarity}(x, \ell^{(3)})$

If $x \approx \ell^{(1)}$: $f_1 \approx \exp(-\frac{0^2}{2\sigma^2}) = 1$

If x is far from $\ell^{(1)}$: $f_1 \approx 0$.

Gamma $=\frac{1}{\sigma^2}$ 是和 C 类似作用的变量，它们越大，越可能 overfitting，越不允许 misclassification.

或者说 decision boundary 的建立与 Gamma 大小有关，Gamma 越大，boundary 的建立取决于离它越近的点，反之亦然.

So, predict 1 when $\theta_0+\theta_1f_1+\theta_2f_2+\theta_3f_3 \geq 0$

But, how to define $\ell^{(1)}, \ell^{(2)}, \ell^{(3)}$?

Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)})$

Choose $l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, \dots \; l^{(m)} = x^{(m)}$

Given example x:

$f_1 = \text{similarity}(x, l^{(1)})$

$f_2 = \text{similarity}(x, l^{(2)})$

For training example $(x^{(i)}, y^{(i)})$

$x^{(i)} \rightarrow \; f_1^{(i)} = \text{sim}(x^{(i)}, l^{(1)})$

$f_2^{(i)} = \text{sim}(x^{(i)}, l^{(2)})$

$\vdots \leftarrow f_i^{(i)} = \text{sim}(x^{(i)}, l^{(i)}) \Leftrightarrow l^{(i)} = x^{(i)}$

$f_m^{(i)} = \text{sim}(x^{(i)}, l^{(m)})$

SVM with kernels

Hypothesis: Given x, compute features $f \in \mathbb{R}^{m+1}$ $(f_0 = 1)$

$\Rightarrow$ predict "$y=1$" if $\theta^T f \geq 0$ $(\theta_0 f_0 + \theta_1 f_1 + \dots + \theta_m f_m)$

f → feature vector

How to get θ?

$$\Rightarrow \min_\theta C \sum_{i=1}^{m} y^{(i)} \text{Cost}_1(\theta^T f^{(i)}) + (1-y^{(i)}) \text{Cost}_0(\theta^T f^{(i)}) + \frac{1}{2} \sum_{j=1}^{\cancel{n}\; m} \theta_j^2 \quad (m=n)$$

$\downarrow$

$\cancel{\theta^T x^{(i)}}$

$x^{(i)}$ 转换到 $f^{(i)}$

低维 → 高维

kernel 的作用在于无需真正把原来的 $x^{(i)}$ 进行从低维到高维的转化（很复杂），也能计算 $x^{(i)}, x^{(j)}$ 之间的 similarity（在高维的情况下）

如果 $m=10000$ 或更大，那么 θ 的维数是10000，在 optimization 时，计算会十分复杂，时间也会很长，所以为了提高效率，在实际情况中对于 regularization term 会有调整.

Note: Large C, Gamma: Lower bias, high variance

Small C, Gamma: Higher bias, low variance

※ 用 10-fold CV 来选 Gamma 和 C，一定要 feature scaling 在用 radial kernel 前.

Multi-class classification

One-vs-all

=> Train k SVMs, one to distinguish $y=i$ from the rest, for $i=1, 2, \cdots, k$, get $\theta^{(1)}, \theta^{(2)}, \cdots, \theta^{(k)}$. Pick class i with largest $(\theta^{(i)})^T x$

Logistic regression vs. SVMs.

n = number of features ($x \in \mathbb{R}^{n+1}$), m = number of training examples

If n is large (relative to m) => $n \geq m$ $n=10000$ $m=10, - 1000$

Use logistic regression or SVM without a kernel (用 linear)

If n is small, m is intermediate. $n=1-1000$, $m=10-10000$

Use SVM with Gaussian kernel.

If n is small, m is large: $n=1-1000$, $m=5{,}0000+$

Create / add more features, then use logistic regression, or SVM without a kernel.

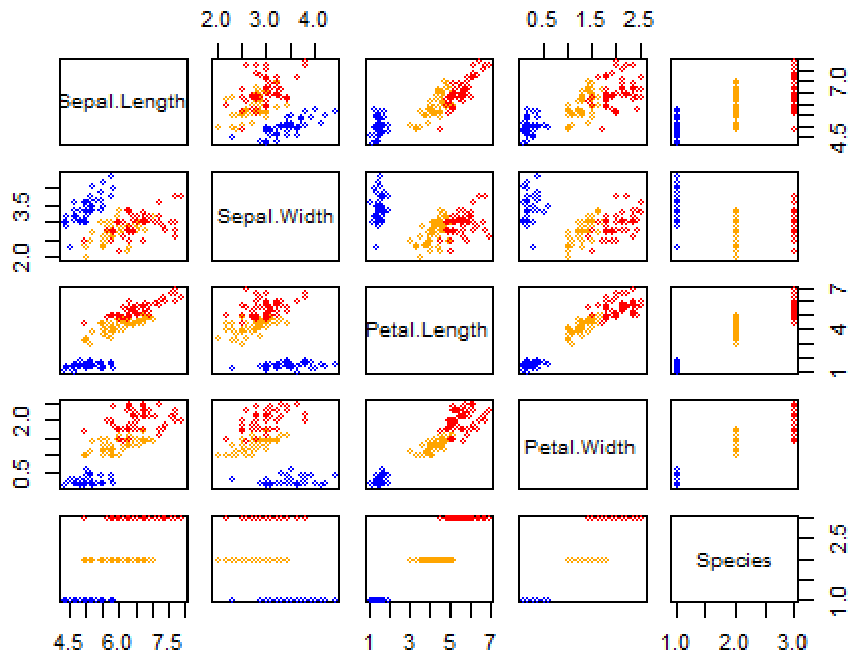
SVM Example

Zijing Gao

```
library(ISLR)

cols <- character(nrow(iris))
cols[] <- "black"

cols[iris$Species == "versicolor"] <- "orange"
cols[iris$Species == "setosa"] <- "blue"
cols[iris$Species == "virginica"] <- "red"
pairs(iris,col=cols, cex = 0.6)
```




```
# setosa is the most separable

#### SVM ####

# train test split
train_idx = sample(nrow(iris), 0.8*nrow(iris))
train = iris[train_idx,]
test = iris[-train_idx,]

# train a SVM model
library(e1071)
```

```r
model = svm(Species~., data = train, probability = TRUE)

summary(model)

##
## Call:
## svm(formula = Species ~ ., data = train, probability = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  46
##
##  ( 18 8 20 )
##
##
## Number of Classes:  3
##
## Levels:
##  setosa versicolor virginica

# prediction with test data
pred = predict(model, subset(test, select = -c(Species)), probability = TRU
E)

# classification error
misClasificError <- mean(pred != test$Species)
print(paste('Accuracy',1-misClasificError))

## [1] "Accuracy 0.966666666666667"

# confusion matrix
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

cm = confusionMatrix(factor(pred), factor(test$Species))
cm

## Confusion Matrix and Statistics
##
##             Reference
```

```
## Prediction   setosa versicolor virginica
##   setosa        10          0         0
##   versicolor     0          9         0
##   virginica      0          1        10
##
## Overall Statistics
##
##                Accuracy : 0.9667
##                  95% CI : (0.8278, 0.9992)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 2.963e-13
##
##                   Kappa : 0.95
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.9000           1.0000
## Specificity                 1.0000            1.0000           0.9500
## Pos Pred Value              1.0000            1.0000           0.9091
## Neg Pred Value              1.0000            0.9524           1.0000
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3333            0.3000           0.3333
## Detection Prevalence        0.3333            0.3000           0.3667
## Balanced Accuracy           1.0000            0.9500           0.9750
```